Model-checking branching-time properties of probabilistic automata and probabilistic one-counter automata

T. Lin

Abstract

This paper studies the problem of model-checking of probabilistic automaton and probabilistic one-counter automata against probabilistic branching-time temporal logics (PCTL and PCTL*). We show that it is undecidable for these problems.

We first show, by reducing to emptiness problem of probabilistic automata, that the model-checking of probabilistic finite automata against branching-time temporal logics are undecidable. And then, for each probabilistic automata, by constructing a probabilistic one-counter automaton with the same behavior as questioned probabilistic automata the undecidability of model-checking problems against branching-time temporal logics are derived, herein.

Keywords: Probabilistic one-counter automata, probabilistic automata, probabilistic one-counter process, Markov decision process, probabilistic branching-time temporal logic, undecidability.

1 Introduction

This paper presents an analysis of model-checking probabilistic branching-time properties of probabilistic automata, in order to study the problem of model-checking probabilistic one-counter automata against probabilistic branching-time temporal logics. Our starting point is that, as is well-known, the class of deterministic one-counter automata is a natural extension of the class of finite-state automata [13]. We can also view the class of probabilistic one-counter automata as a natural extension of the class of probabilistic automata. Herein, if we can show that the questioned problem is undecidable for a restricted one, then the same problem is also undecidable for the more general one. But the inverse is false.

Roughly, probabilistic one-counter automata are probabilistic extension of nondeterministic one-counter automata. A nondeterministic one-counter automaton is just a nondeterministic pushdown automaton with one stack symbol Z except the initial stack-bottom symbol Z_0 which will remain static during the computing. The probabilistic behaviors is placed by adding probabilistic distribution over the set of nondeterministic choices with which when the corresponding nondeterministic one-counter automaton faces. When restricting the input alphabet to contain one symbol, probabilistic one-counter automaton will induce a degenerate case which will be called in this paper the probabilistic one-counter process.

Probabilistic automata, since it was introduced by Rabin [17], has been exhaustively studied from the perspectives of automata theory. For instance, the equivalence problem and reduced form problem were investigated by Carlyle [9] in 1960s. While its cut-point emptiness problem was first shown to be undecidable by Paz in [2] which was reduced indirectly from Post Corresponding Problem [6]. On the other hand, probabilistic automata as a kind of simplest probabilistic computation model, there exists few work relates to formal verification, only a paper [16] dealing with probabilistic timed automata can be found currently. But we have no idea about whether this problem is considered trivial by the community.

Intuitively, a probabilistic automaton consists of a finite set of states, a finite set of input symbols, an initial unit vector, and to each input symbol, a set of driving matrices which are

stochastic are related. The stochastic matrix M_σ for input symbol σ can be interpreted as follows. We assume that there are just n distinct states, then for states q_i, q_j, the probability of the transition which is of the form $q_i \xrightarrow{\sigma} q_j$, denoted as $\mathcal{P}(q_i \xrightarrow{\sigma} q_j)$, is arranged within the (i, j) position of M_σ. In automaton theory, one of the key tasks is to study various properties of probability of accepting a finite word $w = x_1 x_2 \cdots x_m$ for a given probabilistic automaton.

In the literature on probabilistic formal verification, the formal model for probabilistic discrete systems was often depicted by Markov chains [18, 3, 8, 4, 11], which are useful in a number of areas including engineering. Recently, many researchers[3, 4, 11] suggested to take advantage of the more general case of Markov chains, i.e. Markov decision process, to describe certain more complicated formal models with probabilistic behaviors. These points are applicable for probabilistic automata and probabilistic one-counter automata.

As previously stated, Markov chains are wildly used in depicting probabilistic discrete systems. But unfortunately, it describes only stochastic, but can not describe controlled behaviors (nondeterministic choice) which Markov decision process can. Intuitively, Markov decision process consists of a countable set of states, a finite set of actions, an initial probabilistic vector ι_{init} over the set of states, and a probabilistic function to each action a, for a state q_i, assigning a value $\mathcal{P}(q_i \xrightarrow{a} q_j) \in [0, 1]$ with $\sum_{q_j} \mathcal{P}(q_i \xrightarrow{a} q_j) \in \{0, 1\}$. In exoteric languages, the dynamics of the system begins in an initial state s such that $\iota_{init}(s) > 0$, then it opts nondeterministically for an enabled action a [1] for state s, followed by changing its state to s' with probability $\mathcal{P}(s \xrightarrow{a} s')$. Note in advance that we will replace the transition $s \xrightarrow{a} s'$ by $(s, a, s') \in S \times Act \times S$ where S is the state set and Act is its action set. Then, this kind of procedure will continue infinitely.

Referring to branching-time temporal logics, we often mean the probabilistic version of branching-time temporal logics. In the paper, two various kinds of such logics will be involved, i.e., the PCTL and PCTL* which are probabilistic extensions of well-known branching-time CTL and CTL* respectively. In such logics, the universal $\forall$ and existential $\exists$ path quantifiers of CTL and CTL* are replaced by probabilistic operator $\mathcal{P}_{\bowtie t}(\varphi)$, saying that the probability of all runs satisfying φ is $\bowtie$-related to t, where $t \in \{>, \geq\}$,[2] $t \in [0, 1]$ is a real and φ is a path formula of PCTL or PCTL*.

We consider the decidability of model-checking probabilistic branching-time properties of such models. Specifically, we establish firstly the following result.

Theorem 1.1. *The model-checking problem of probabilistic automata against PCTL (PCTL*) is undecidable.*

With the Theorem 1.1 in place, we construct, for each probabilistic finite automaton, a probabilistic one-counter automaton. Herein, we show next the main result

Theorem 1.2. *The model-checking of probabilistic branching-time properties of probabilistic one-counter automata is undecidable.*

If let the input alphabet of probabilistic one-counter automata be of only one symbol, this special case of probabilistic one-counter automata will be called 'probabilistic one-counter process'. Given a probabilistic one-counter process, it induces a infinite Markov chain. Its model-checking problem remains open in this paper, because it seems that the method to Theorem 1.2 is not applicable for this case. We conjecture it is decidable, based on that it is not possible to encode "Post Corresponding Problem" to its transition rules.

The remainder of the paper is structured in the following way. In Section 2, we recall some necessary definitions. After establishing some necessary technical lemmas, we prove Theorem 1.1 in Section 3. The proof of Theorem 1.2 is put into Section 4. Finally, we draw some conclusions in the last Section.

[1] In this case, it must satisfy $\sum_{s'} \mathcal{P}(s \xrightarrow{a} s') = 1$, because the action a satisfying $\sum_{s'} \mathcal{P}(s \xrightarrow{a} s') = 0$ is not enabled.

[2] Generally, the set of comparison relations is $\{>, <, =, \geq, \leq\}$, but $\{>, \geq\}$ will suffice for our context.

2 Definitions

Throughout the paper, for any set S (finite or infinite), $|S|$ denotes the cardinality of S, and ω denotes either the set $\{0, 1, 2, \cdots\}$, or $|\omega|$, depending on the contexts. Σ denotes the non-empty finite alphabet, Σ^* denotes the set of all finite words (including empty word ϵ) over Σ, and $\Sigma^+ = \Sigma^* \setminus \{\epsilon\}$. Let w be a word in Σ^*, then $|w|$ will denote the length of w. For two words $w_1, w_2 \in \Sigma^*$, their conjunction is denoted by $w_1 w_2$. For some information about probability theory, the reader is refereed to either [1] or [14, 15], depending on individual preferences. Meanwhile, the reader can find useful information about model-checking in [7].

Markov Decision Process

Markov decision processes are probabilistic models which are useful in a number of areas including engineering. The one presented here are adapted from [3]. For more details, please consult [3, 5].

Definition 2.1. *A (discrete) Markov decision process is a tuple $\widetilde{\mathcal{M}} = (S, Act, \mathcal{P}, \iota_{\text{init}})$ where S is a countable set of states, Act a set of actions, $\mathcal{P}$: $S \times Act \times S \to [0, 1]$ the transition probability function such that for all $s \in S$ and $a \in Act$*

$$\sum_{s' \in S} \mathcal{P}(s, a, s') \quad \in \quad \{0, 1\},$$

and $\iota_{\text{init}} : S \to [0, 1]$ is the initial distribution such that $\sum_{s \in S} \iota_{\text{init}}(s) = 1$. An action $a \in Act$ is enabled in state s if and only if $\sum_{s' \in S} \mathcal{P}(s, a, s') = 1$.

Remark 1. *Denote the set of enabled actions in state s by $A(s)$, i.e.,*

$$A(s) \quad = \quad \left\{ a \in Act \,|\, \sum_{s' \in S} \mathcal{P}(s, a, s') = 1 \right\}$$

It is required that $|A(s)| > 0$ for any $s \in S$. When $|A(s)| = 1$ for any $s \in S$, then it is clear that an MDP is an Markov chain.

Definition 2.2. *A path in $\widetilde{\mathcal{M}} = (S, Act, \mathcal{P}, \iota_{\text{init}})$ is either a finite or infinite sequence of the form*

$$\sigma_w \quad = \quad s_0 \xrightarrow{a_1} s_1 \xrightarrow{a_2} s_2 \xrightarrow{a_3} \cdots \xrightarrow{a_{|w|}} s_{|w|},$$

depending on whether $|w| < \omega$ or $|w| = \omega$, where $w = a_1 a_2 \cdots a_{|w|}$, $s_i \in S$ and $a_i \in Act$ such that

$$\mathcal{P}(s_i, a_{i+1}, s_{i+1}) > 0, \qquad \text{and} \sum_{s'_{i+1} \in S} \mathcal{P}(s_i, a_{i+1}, s'_{i+1}) = 1.$$

Further, we use $Paths(s)$ and $IPaths(s)$ to denote respectively the set of finite paths and infinite paths that begin in s. Also, we let $Paths(\widetilde{\mathcal{M}}) = \bigcup_{s \in S} Paths(s)$ and $IPaths(\widetilde{\mathcal{M}}) = \bigcup_{s \in S} IPaths(s)$.

It is convenient to recall the definition of Markov chains. Roughly, Markov chains are probabilistic transition systems which are accepted [3] as the most popular operational model for the evaluation of performance and dependability of information-processing systems. For convenience, the following definition is re-stated from [18].

Definition 2.3. *A (discrete) Markov chain is a triple $\mathcal{M} = (S, \delta, \mathcal{P})$ where S is a finite or countably infinite set of states, $\delta \subseteq S \times S$ is a transition relation such that for each $s \in S$ there exits $t \in S$ such that $(s, t) \in \delta$, and $\mathcal{P}$ is a function from domain δ to range $(0, 1]$ which to each transition $(s, t) \in \delta$ assigns its probability $\mathcal{P}(s, t)$ such that $\sum_{(s,t) \in \delta} \mathcal{P}(s, t) = 1$ for all $s \in S$.*

A path in $\mathcal{M}$ is a finite or infinite sequence of states of S: $\rho = s_0 s_1 \cdots$ such that $(s_i, s_{i+1}) \in \delta$ for each i. A run of $\mathcal{M}$ is an infinite path. We denote the set of all runs in $\mathcal{M}$ by Run, and $Run(\rho')$ to denote the set of all runs starting with a given finite path ρ'. Let ρ be a given run, then $\rho(i)$ denotes the state s_i of ρ, and ρ_i the run $s_i s_{i+1} \cdots$. In this way, it is clear that $\rho_0 = \rho$. Further, a state s' is *reachable* from a state s if there is a *finite path* starting in s and ending at s'.

For each $s \in S$, $(Run(s), \mathcal{F}, \mathcal{P})$ is a probability space, where $\mathcal{F}$ is the ρ-field generated by all *basic cylinders* $Run(\rho)$ where ρ is a finite path initiating from s, and $\mathcal{P} : \mathcal{F} \to [0, 1]$ is the unique probability measure such that $\mathcal{P}(Run(\rho)) = \prod_{1 \le i \le |\rho|} \mathcal{P}(s_{i-1}, s_i)$ where $\rho = s_0 s_1 \cdots s_{|\rho|}$.

Remark 2. *Generally, MDPs are not augmented with a unique probability measure because of the nondeterminism when choosing an action in $A(s)$. We refer the reader to monograph [3] for some more detailed examples. Among many approaches to resolve the nondeterminism of an MDP, the usual one is that we can introduce a scheduler.*

Definition 2.4 (Cf. Definition 10.91 [3])**.** *Let $\widetilde{\mathcal{M}} = (S, Act, \mathcal{P}, \iota_{\text{init}})$ be an MDP. A scheduler for $\widetilde{\mathcal{M}}$ is a function $\mathfrak{S}$: $S^+ \to Act$ such that $\mathfrak{S}(s_0 s_1 \cdots s_n) \in A(s_n)$ for all $s_0 s_1 \cdots s_n \in S^+$. Now we can call the path*

$$w \quad = \quad s_0 \xrightarrow{a_1} s_1 \xrightarrow{a_2} s_2 \xrightarrow{a_3} \cdots \xrightarrow{a_{|w|}} s_{|w|}$$

a $\mathfrak{S}$-path if $a_i = \mathfrak{S}(s_0 \cdots s_{i-1})$ for all $i > 0$.

Now, we can reason about the probability of sets of $\mathfrak{S}$-paths, because it is clear to see that a scheduler $\mathfrak{S}$ induces a Markov chain. More formally, we have

Definition 2.5 (Cf. Definition 10.92 [3])**.** *Let $\widetilde{\mathcal{M}} = (S, Act, \mathcal{P}, \iota_{\text{init}})$ be an MDP and $\mathfrak{S}$ a scheduler on $\widetilde{\mathcal{M}}$. The Markov chain $\mathcal{M}_{\mathfrak{S}}$ induced by $\mathfrak{S}$ is a tuple*

$$\mathcal{M}_{\mathfrak{S}} \quad = \quad (S^+, \delta, \mathcal{P}_{\mathfrak{S}})$$

where $\delta \subseteq S^+ \times S^+$. For any $(w, ws_{n+1}) \in \delta$ where $w = s_0 s_1 \cdots s_n \in S^+$:

$$\mathcal{P}_{\mathfrak{S}}(w, ws_{n+1}) \quad = \quad \mathcal{P}(s_n, \mathfrak{S}(w), s_{n+1}).$$

From the above definition, we clearly see that $\mathcal{M}_{\mathfrak{S}}$ is an infinite Markov chain (Even if the MDP $\widetilde{\mathcal{M}}$ is finite). In this case, an infinite $\mathfrak{S}$-path is called a $\mathfrak{S}$-run. Further denote the set of $\mathfrak{S}$-runs that start s by $\mathfrak{S}Runs(s)$. That is

$$\mathfrak{S}Runs(s) \quad = \{\rho \,|\, \rho \text{ is a } \mathfrak{S}\text{-path that starts from } s\}$$

Probabilistic Computational Tree Logic

The logic PCTL

The logic PCTL, which is slightly different with the one in this paper, was originally introduced by Hansson et al. in [8], where the corresponding model-checking problem has been focused mainly on finite-state Markov chains.

Let AP be a fixed set of atomic propositions. Formally, the syntax of PCTL is defined by

$$\begin{aligned} \Phi &::= p \mid \neg\Phi \mid \Phi_1 \wedge \Phi_2 \mid \mathcal{P}_{\bowtie r}(\varphi) \\ \varphi &::= \mathbf{X}\Phi \mid \Phi_1 \mathbf{U} \Phi_2 \end{aligned}$$

where Φ and φ denote the state formula and path formula respectively; $p \in AP$ is an atomic proposition, $\bowtie \in \{>, \ge\}$, r is a real with $0 \le r \le 1$. The symbol **true** is the abbreviation of always true.

Let $\mathcal{M} = (S, \delta, \mathcal{P})$ be a Markov chain and $\nu : AP \to 2^S$ an assignment. Then the semantics of PCTL, over $\mathcal{M}$, is given by the following rules

$$
\begin{array}{lll}
\mathcal{M}, s \models^{\nu} \mathbf{true} & \text{for any } s \in S, & \\
\mathcal{M}, s \models^{\nu} p & \Leftrightarrow & s \in \nu(p), \\
\mathcal{M}, s \models^{\nu} \neg\Phi & \Leftrightarrow & \mathcal{M}, s \not\models^{\nu} \Phi, \\
\mathcal{M}, s \models^{\nu} \Phi_1 \wedge \Phi_2 & \Leftrightarrow & \mathcal{M}, s \models^{\nu} \Phi_1 \text{ and } \mathcal{M}, s \models^{\nu} \Phi_2, \\
\mathcal{M}, s \models^{\nu} \mathcal{P}_{\bowtie r}(\varphi) & \Leftrightarrow & \mathcal{P}(\{\rho \in Run(s) : \mathcal{M}, \rho \models^{\nu} \varphi\}) \bowtie r, \\
 & & \\
\mathcal{M}, \rho \models^{\nu} \mathbf{X}\Phi & \Leftrightarrow & \mathcal{M}, \rho(1) \models^{\nu} \Phi, \\
\mathcal{M}, \rho \models^{\nu} \Phi_1 \mathbf{U} \Phi_2 & \Leftrightarrow & \exists k \geq 0.(\mathcal{M}, \rho_k \models^{\nu} \Phi_2 \wedge \forall j < k.(\mathcal{M}, \rho_j \models^{\nu} \Phi_1)).
\end{array}
$$

The logic PCTL*

The logic PCTL*, containing PCTL as a sublogic, is a kind of extensions of PCTL. The syntax of state formula is the same as for PCTL, while that of path formula is given by

$$\varphi ::= \Phi \,|\, \neg\varphi \,|\, \varphi_1 \wedge \varphi_2 \,|\, \mathbf{X}\,\varphi \,|\, \varphi_1 \,\mathbf{U}\, \varphi_2.$$

The semantics of PCTL* path formulas are defined, over $\mathcal{M}$, as follows

$$
\begin{array}{lll}
\mathcal{M}, \rho \models^{\nu} \Phi & \Leftrightarrow & \mathcal{M}, \rho(0) \models^{\nu} \Phi, \\
\mathcal{M}, \rho \models^{\nu} \neg\varphi, & \Leftrightarrow & \mathcal{M}, \rho \not\models^{\nu} \varphi \\
\mathcal{M}, \rho \models^{\nu} \varphi_1 \wedge \varphi_2 & \Leftrightarrow & \mathcal{M}, \rho \models^{\nu} \varphi_1 \text{ and } \mathcal{M}, \rho \models^{\nu} \varphi_2, \\
\mathcal{M}, \rho \models^{\nu} \mathbf{X}\,\varphi & \Leftrightarrow & \mathcal{M}, \rho_1 \models^{\nu} \varphi \\
\mathcal{M}, \rho \models^{\nu} \varphi_1 \,\mathbf{U}\, \varphi_2 & \Leftrightarrow & \exists k \geq 0.(\mathcal{M}, \rho_k \models^{\nu} \Phi_2 \wedge \forall j < k.(\mathcal{M}, \rho_j \models^{\nu} \Phi_1))
\end{array}
$$

where ν: $AP \to 2^S$ is an assignment.

MDPs as Semantic Models of PCTL/PCTL*

Thus far, we have not mentioned how to specify important properties of finite MDPs by means of PCTL/PCTL*, but only given an interpretation of PCTL/PCTL* over a Markov chain. Without doubt, to give a rational interpretation of PCTL/PCTL* over an MDP is more complicated due to nondeterminism of MDPs, see Remark 2. We quote the common way presented in [3].

Let AP be a set of atomic propositions, $\widetilde{\mathcal{M}} = (S, Act, \mathcal{P}, \iota_{\text{init}})$ an MDP, and ν an assignment $AP \to 2^S$. The way of interpreting PCTL/PCTL* over MDPs is, to some large extent, the same as for Markov chains, except the probabilistic operator $\mathcal{P}_{\bowtie\lambda}(\varphi)$ where φ is a path formula of PCTL/PCTL*. The specific approach is given as follows

$$
\begin{array}{lll}
\widetilde{\mathcal{M}}, s \models^{\nu} \mathbf{true} & \text{for any } s \in S; & \\
\widetilde{\mathcal{M}}, s \models^{\nu} p & \Leftrightarrow & \text{if } p \in \nu(s); \\
\widetilde{\mathcal{M}}, s \models^{\nu} \neg\Phi & \Leftrightarrow & \widetilde{\mathcal{M}}, s \not\models^{\nu} \Phi; \\
\widetilde{\mathcal{M}}, s \models^{\nu} \Phi_1 \wedge \Phi_2 & \Leftrightarrow & \widetilde{\mathcal{M}}, s \models^{\nu} \Phi_1 \text{ and } \widetilde{\mathcal{M}}, s \models^{\nu} \Phi_2;
\end{array}
$$

$$
\begin{array}{lll}
\widetilde{\mathcal{M}}, s \models^{\nu} \mathcal{P}_{\bowtie r}(\varphi) & \Leftrightarrow & \forall \mathfrak{S} : \ \mathcal{P}_{\mathfrak{S}}(\{\rho \in \mathfrak{S}Runs(s) \ : \ \widetilde{\mathcal{M}}, \rho \models^{\nu} \varphi\}) \bowtie r \\
\widetilde{\mathcal{M}}, \iota_{init} \models^{\nu} \mathcal{P}_{\bowtie r}(\varphi) & \Leftrightarrow & \forall \mathfrak{S} : \displaystyle\sum_{s:\, \iota_{init}(s) > 0} \mathcal{P}_{\mathfrak{S}}(\{\rho \in \mathfrak{S}Runs(s) \ : \ \widetilde{\mathcal{M}}, \rho \models^{\nu} \varphi\}) \bowtie r
\end{array}
$$

with that the semantics of path formula is exactly the same as for PCTL/PCTL* interpreted over Markov chains.

3 Proof of Theorem 1.1

Probabilistic Automata

Let us review the standard definition of probabilistic automata, first.

Definition 3.1 (Adaptation from [17]). *A probabilistic automaton (p.a.) over an input alphabet Σ is a 4-tuple $\mathcal{A} = (Q, \{M_a | a \in \Sigma\}, \pi, F)$ where Q is a finite set of states, M_a a stochastic matrix, π, a row vector, is an initial distribution over Q, and $F \subseteq Q$ is the accepting set.*

The main behavior of $\mathcal{A}$ is that, for each word $w = a_1 a_2 \cdots a_n \in \Sigma^*$, it induces a word function $\mathcal{P}_{\mathcal{A}}(w)$, which is given by

$$\begin{aligned} \mathcal{P}_{\mathcal{A}}(w) &\triangleq \pi M_{a_1} M_{a_2} \cdots M_{a_n} \eta_F \\ &= \pi M_w \eta_F \end{aligned}$$

where $\eta_F = (e_{j1})_{m \times 1}$ (assuming that $|Q| = m$) such that $e_{j1} = 1$ if $q_j \in F$, and $e_{j1} = 0$ otherwise.

Let λ, which will be called a cut-point, be an arbitrary real number satisfying that $0 \leq \lambda \leq 1$, we define the languages recognized by $\mathcal{A}$ with λ in the following ways

$$\begin{aligned} L_{>\lambda}(\mathcal{A}) &= \{w \in \Sigma^* \,|\, \mathcal{P}_{\mathcal{A}}(w) > \lambda\} \\ L_{\geq\lambda}(\mathcal{A}) &= \{w \in \Sigma^* \,|\, \mathcal{P}_{\mathcal{A}}(w) \geq \lambda\}, \end{aligned}$$

in which the first and second are called the strict and non-strict languages recognized by $\mathcal{A}$ respectively. With these notions in place, the problems of strict and non-strict emptiness are defined to ask whether the sets $L_{>\lambda}(\mathcal{A})$ and $L_{\geq\lambda}(\mathcal{A})$ respectively are empty or not. Paz showed in his book [2] that both problems are undecidable.

Proposition 3.1. *(Adaptation from Theorem 6.17 [2]) Let $\mathcal{A} = (Q, \{M_a | a \in \Sigma\}, \pi, F)$ be an arbitrary probabilistic automaton, and λ: $0 \leq \lambda \leq 1$ an arbitrary cutpoint. Then, it is undecidable*

(1) (strict emptiness.) whether the set $\{w \in \Sigma^* \,|\, \mathcal{P}_{\mathcal{A}}(w) > \lambda\}$ is empty;

(2) (non-strict emptiness.) whether the set $\{w \in \Sigma^* \,|\, \mathcal{P}_{\mathcal{A}}(w) \geq \lambda\}$ is empty.

Now we re-state the definition of probabilistic automata in terms of the langauge of Markov decision process. Let $\mathcal{A} = (Q, \{M_a | a \in \Sigma\}, \pi, F)$ be a probabilistic automaton. View $S = Q$ and $Act = \Sigma$, we can write the stochastic matrix M_a as follows

$$M_a = \begin{pmatrix} p_{11} & p_{12} & \cdots & p_{1m} \\ p_{21} & p_{22} & \cdots & p_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ p_{m1} & p_{m2} & \cdots & p_{mm} \end{pmatrix}$$

where $p_{ij} \in [0, 1]$ with $\sum_{j=1}^{n} p_{ij} = 1$ for all i. The intuitive mean is that when $\mathcal{A}$ is in state q_i, with the input symbol a (The action in the MDP), it changes into state q_j with probability p_{ij}. Hence the probability function in the MDP can be written as

$$\mathcal{P} = \{M_a(i, j) \,|\, a \in Act, 1 \leq i, j \leq |S|\}.$$

Assume next that the input alphabet $\Sigma = \{a_1, a_2, \cdots, a_n\}$, i.e., the Act for an MDP, and π corresponds to ι_{init}. Then, a probabilistic automaton $\mathcal{A}$ is just an MDP [3] $\widetilde{\mathcal{M}}_{\mathcal{A}} = (S, Act, \mathcal{P}, \iota_{\text{init}})$

[3] Strictly speaking, $\mathcal{A}$ should be considered as an MDP with additional accepting set F.

with $S = Q$, $Act = \Sigma$, $\mathcal{P} = \{M_a(i,j) \,|\, a \in Act, 1 \leq i,j \leq |S|\}$, and $\iota_{\text{init}} = \pi$. It is worth of noting that

$$\begin{aligned} A(s) &= \left\{ a \in Act \,|\, \sum_{s' \in S} \mathcal{P}(s,a,s') = 1 \right\} \\ &= Act \ (= \Sigma) \end{aligned}$$

for any $s \in S$.

3.1 Proof of Theorem 1.1

We show an important technical lemma, firstly.

Lemma 3.1. *Let $\mathcal{A} = (Q, \{M_a | a \in \Sigma\}, \pi, F)$ be a probabilistic automaton, $\widetilde{\mathcal{M}}_{\mathcal{A}} = (S, Act, \mathcal{P}, \iota_{init})$ an MDP induced by $\mathcal{A}$ where $S = Q, Act = \Sigma$, $\mathcal{P} = \{M_a(i,j) \,|\, a \in Act, 1 \leq i,j \leq |S|\}$, and $\iota_{init} = \pi$. Further, let $\widetilde{Paths(s)}$ be the set of finite $\mathfrak{S}$-paths of $\widetilde{\mathcal{M}}_{\mathcal{A}}$, given by*

$$\widetilde{Paths(s)} = \left\{ \rho = s \xrightarrow{\mathfrak{S}(s)} s_1 \xrightarrow{\mathfrak{S}(ss_1)} \cdots \xrightarrow{\mathfrak{S}(ss_1 \cdots s_{n-2})} s_{n-1} \xrightarrow{\mathfrak{S}(ss_1 \cdots s_{n-1})} s_n \,|\, s_n \in F \right\}$$

and

$$\mathcal{P}_{\mathfrak{S}}(\widetilde{Paths(s)}) = \sum_{\rho \in \widetilde{Paths(s)}} \mathcal{P}(s, \mathfrak{S}(s), s_1)\mathcal{P}(s_1, \mathfrak{S}(ss_1), s_2) \cdots \mathcal{P}(s_{n-1}, \mathfrak{S}(ss_1 \cdots s_{n-1}), s_n)$$

It can be claimed that, given an arbitrary finite word $w = a_1 a_2 \cdots a_n \in \Sigma^*$, there exists a scheduler $\mathfrak{S}_w$ such that

$$\mathcal{P}_{\mathcal{A}}(w) = \sum_{s \in S} \iota_{init}(s) \cdot \mathcal{P}_{\mathfrak{S}_w}(\widetilde{Paths(s)}) \tag{1}$$

Proof. The desirable scheduler $\mathfrak{S}_w$ is defined inductively as follows. For convenience, we assume that $|Q| = m$. Given the finite word $w = a_1 a_2 \cdots a_n \in \Sigma^*$, let

$$\begin{aligned} \tau_1 &= \iota_{init} \cdot M_{a_1}; \\ \tau_2 &= \tau_1 \cdot M_{a_2}; \\ &\vdots \\ \tau_n &= \tau_{n-1} \cdot M_{a_n} \end{aligned}$$

Then, define

$$\begin{aligned} &\mathfrak{S}_w(s_1) = a_1 \quad \text{for any } s_1 \in S \text{ with } \tau_1(s_1) > 0; \\ &\mathfrak{S}_w(s_1 s_2) = a_2 \quad \text{for any } s_2 \in S \text{ with } \tau_2(s_2) > 0; \\ &\mathfrak{S}_w(s_0 s_1 \cdots s_i) = a_{i+1} \quad \text{if } \mathfrak{S}_w(s_0 s_1 \cdots s_{i-1}) \text{ is defined, and for any } s_i \in S \text{ with } \tau_i(s_i) > 0 \end{aligned}$$

It is clear that $\mathfrak{S}_w$ is indeed a function from S^+ to Act, because for any $s_0 s_1 \cdots s_i \in S^+$, there is at most only one element in Act corresponding to $\mathfrak{S}_w(s_0 s_1 \cdots s_i)$.

We then need to to show that Eq. (1) is true. To do so, recall the definitions of $\mathfrak{S}_w$, $\widetilde{Paths(s)}$ as well as $\mathcal{P}_{\mathfrak{S}_w}(\widetilde{Paths(s)})$. It is clear that the right hand of Eq. (1) is the overall probability of finite $\mathfrak{S}_w$-paths which is of length $|w|$, starting from ι_{init} then successively applying actions $a_1, a_2, \cdots, a_n$, finally reaching a state $s_f \in F$. It is just the definition of $\mathcal{P}_{\mathcal{A}}(w)$. □

Let $\mathcal{P}_{\mathfrak{S}}(\widetilde{\mathcal{M}}_{\mathcal{A}}, \iota_{init} \models^{\nu} \varphi)$ further denote the probability that under the schuller $\mathfrak{S}$, all $\mathfrak{S}$-runs which start from ι_{init}, satisfying the path formula φ. Now, we depict an assignment ν and construct one PCTL formula φ_w such that

$$\mathcal{P}_{\mathcal{A}}(w) \quad = \quad \mathcal{P}_{\mathfrak{S}_w}(\widetilde{\mathcal{M}}_{\mathcal{A}}, \iota_{init} \models^{\nu} \varphi_w)$$

for a given finite word $w = a_1 a_2 \cdots a_n \in \Sigma^*$.

Definition 3.2. *The assignment ν is defined as follows. Let $AP = \Sigma \cup \{Accept\}$ be the atomic propositions. For any $s, s' \in S$ and $a \in \Sigma$, if $\mathcal{P}(s, a, s') > 0$ (i.e., the (s, s') index of M_a is larger than 0), then $a \in \nu(s')$. Additionally, $\nu(s) = Accept$ if $s \in F$.*

We construct a PCTL formulae which is useful in the later. Given a finite word $w = a_1 \cdots a_n \in \Sigma^*$. Let φ_w be defined inductively as follows.

$$\begin{aligned}
\varphi_w &\triangleq \mathbf{true}\mathbf{U}\mathcal{P}_{>0}(\mathbf{X}\varphi_1) \\
\varphi_1 &\triangleq a_1 \wedge \mathcal{P}_{>0}(\mathbf{X}\varphi_2) \\
&\vdots \\
\varphi_{n-2} &\triangleq a_{n-2} \wedge \mathcal{P}_{>0}(\mathbf{X}\varphi_{n-1}) \\
\varphi_{n-1} &\triangleq a_{n-1} \wedge \mathcal{P}_{>0}(\mathbf{X}\varphi_n) \\
\varphi_n &\triangleq a_n \wedge Accept.
\end{aligned}$$

Lemma 3.2. *Let $\mathcal{A} = (Q, \{M_a | a \in \Sigma\}, \pi, F)$ be a probabilistic automaton and $\widetilde{\mathcal{M}}_{\mathcal{A}} = (S, Act, \mathcal{P}, \iota_{init})$ an MDP induced by $\mathcal{A}$. Given an arbitrary finite word $w = a_1 \cdots a_n \in \Sigma^*$. Then we have the following*

$$\sum_{s \in S} \iota_{init}(s) \cdot \mathcal{P}_{\mathfrak{S}_w}(\widetilde{Paths(s)}) \quad = \quad \mathcal{P}_{\mathfrak{S}_w}(\widetilde{\mathcal{M}}_{\mathcal{A}}, \iota_{init} \models^{\nu} \varphi_w)$$

where ν is as in Definition 3.2.

Proof. We assume the finite $\mathfrak{S}_w$-path ρ of length $|w|$ is the following

$$\rho \quad = \quad s_0 \xrightarrow{a_1} s_1 \xrightarrow{a_2} s_2 \xrightarrow{a_3} \cdots \xrightarrow{a_{n-1}} s_{n-1} \xrightarrow{a_n} s_n$$

It is clear that if $\widetilde{\mathcal{M}}_{\mathcal{A}}, \rho \models^{\nu} \varphi_w$, then firstly, $s_n \models^{\nu} a_n \wedge Accept$ should be stratified by the assignment ν. Denoting $\varphi_n = a_n \wedge Accept$, as $\mathcal{P}(s_{n-1}, a_n, s_n) > 0$ and $s_n \models^{\nu} \varphi_n$, this gives $s_{n-1} \models^{\nu} \mathcal{P}_{>0}(\mathbf{X}\varphi_n))$. Note also that, $s_{n-1} \models^{\nu} a_{n-1}$, we get $s_{n-1} \models^{\nu} a_{n-1} \wedge \mathcal{P}_{>0}(\mathbf{X}\varphi_n)$. We can further denote the formula $a_{n-1} \wedge \mathcal{P}_{>0}(\mathbf{X}\varphi_n)$ by φ_{n-1}.

Inductively, by the above analysis, we can get that $s_i \models^{\nu} a_i \wedge \mathcal{P}_{>0}(\mathbf{X}\varphi_{i+1})$ for all $1 \leq i \leq n-1$.

Now, it is clear that $s_0 \models^{\nu} \mathbf{X}\varphi_1$ if $\mathcal{P}(s_0, a_1, s_1) > 0$, which gives $s_0 \models^{\nu} \mathcal{P}_{>0}(\mathbf{X}\varphi_1)$. But the formula $\mathcal{P}_{>0}(\mathbf{X}\varphi_1)$ is not a path formula, rather than a state formula. Recall the semantics of $s \models^{\nu} \mathbf{true}\mathbf{U}\Phi$ (Here, s should be considered as a finite path of length 1). It is clear that $s_0 \models^{\nu} \mathcal{P}_{>0}(\mathbf{X}\varphi_1)$ is equivalent to $s_0 \models^{\nu} \mathbf{true}\mathbf{U}\mathcal{P}_{>0}(\mathbf{X}\varphi_1)$, which is a PCTL formula. It is useful to denote $\mathbf{true}\mathbf{U}\mathcal{P}_{>0}(\mathbf{X}\varphi_1)$ by φ_w.

It is clear that the probability of all of the $\mathfrak{S}_w$-paths satisfying φ_w, i.e., $\mathcal{P}_{\mathfrak{S}_w}(\widetilde{\mathcal{M}}_{\mathcal{A}}, \iota_{init} \models^{\nu} \varphi_w)$, is just $\sum_{s \in S} \iota_{init}(s) \cdot \mathcal{P}_{\mathfrak{S}_w}(\widetilde{Paths(s)})$. □

We present the proof of the Theorem 1.1 as follows.

Proof of Theorem 1.1. We show it by contradiction. Assume Theorem 1.1 is false, then for any probabilistic automaton $\mathcal{A}$, for any finite word $w = a_1 a_2 \cdots a_n \in \Sigma^*$, for any $p \in [0, 1]$, it is decidable that

$$\widetilde{\mathcal{M}}_{\mathcal{A}}, \iota_{init} \models^{\nu} \mathcal{P}_{>p}(\varphi_w)$$

i.e., it is decidable that

$$\mathcal{P}_{\mathfrak{S}}\left(\widetilde{\mathcal{M}}_{\mathcal{A}}, \iota_{init} \models^{\nu} \varphi_w\right) \quad > \quad p \qquad \text{for all } \mathfrak{S},$$

which includes the case that it is decidable whether or not

$$\mathcal{P}_{\mathfrak{S}_w}\left(\widetilde{\mathcal{M}}_{\mathcal{A}}, \iota_{init} \models^{\nu} \varphi_w\right) \quad > \quad p$$

This, by Lemma 3.1 and Lemma 3.2, leads to an algorithm to decide whether $\mathcal{P}_{\mathcal{A}}(w) > p$, contracting to the cause 1 of Proposition 3.1. □

Remark 3. *Note that, the proof of undecidability of model-checking PCTL path formula $\mathcal{P}_{\geq p}(\varphi)$ to probabilistic automata is similar. Note also that PCTL is a sublogic of PCTL*. Hence, Theorem 1.1 is also true for PCTL*.*

4 Model-checking probabilistic one-counter automata

Probabilistic one-counter automata

Let us first recall the standard definition of probabilistic one-counter automata. Note that the probabilistic one-counter automata are probabilistic extension of nondeterministic one-counter automata. Note also that the nondeterministic one-counter automata are nondeterministic extension of deterministic one-counter automata, whose description can be found in [10, 13].

A probabilistic one-counter automaton R is described by a 8-tuple $R = (K, \Sigma, \Gamma, \delta, Z_0, S, F, \mathcal{P})$, where

- K is a nonempty finite set (of states).
- Σ is a nonempty finite set (of inputs).
- Γ is a finite nonempty set containing only two stack symbols.
- δ is a mapping from $K \times (\Sigma \cup \{\epsilon\}) \times \Gamma$ into the finite subsets of $K \times \Gamma^*$.
- Z_0 is an element of Γ, the start stack symbol.
- q is in K, the set of start states.
- F is a subset of K, the set of *final* states.
- $\mathcal{P}$ is a probabilistic distribution from δ to range $(0, 1]$ stratifying that

$$\sum_i \mathcal{P}((q, Z) \xrightarrow{a} (p_i, \gamma_i)) \quad = \quad 1 \qquad \text{if } \delta(q, a, Z) = \{(p_1, \gamma_1), \cdots, (p_l, \gamma_l)\}$$

A *configuration* of M is described by (q, n) where $q \in K$, and n is the number of Z in the stack.

With the standard definition in place, we show the following

Lemma 4.1. *Let $\mathcal{A} = (Q, \{M_a | a \in \Sigma\}, \pi, F)$ be any arbitrary probabilistic automaton. Then there exists a probabilistic one-counter automaton R such that for any word $w \in \Sigma^*$, the probability of accepting w both for $\mathcal{A}$ and for R are equal:*

$$\mathcal{P}_{\mathcal{A}}(w) \quad = \quad \mathcal{P}_R(w).$$

Proof. Construct $R = (K, \Sigma, \Gamma, \delta, Z_0, q_0, F', \mathcal{P})$ as follows.

First, let $K = Q \cup \{q_0\}$ with assumption that $q_0 \notin Q$, and $F' = F$ (of course, the input alphabet of $\mathcal{A}$ and R are identical). For any $s \in Q$ such that $\pi(s) > 0$ we add the following rule into δ:

$$(q_0, Z_0) \xrightarrow{\epsilon} (s, Z) \qquad \text{with the probability } \pi(s)$$

Next, we define the remainder of transition rules into δ. For any M_a,

$$(q_i, Z) \xrightarrow{a} (q_j, Z)$$

is in δ with $\mathcal{P}((q_i, Z) \xrightarrow{a} (q_j, Z)) = M_a(i, j)$. It is obvious that

$$\sum_j \mathcal{P}((q_i, Z) \xrightarrow{a} (q_j, Z)) = \sum_j M_a(i, j) = 1$$

The above constructing leads to another 'probabilistic automaton' with an additional state (q_0, Z_0). And the matrices are $\{M'_\epsilon, \} \cup \{M'_a | a \in \Sigma\}$, where

$$M'_\epsilon = \begin{pmatrix} 0 & \pi \\ \mathbf{0} & I \end{pmatrix} \quad M'_a = \begin{pmatrix} 0 & \pi \\ \mathbf{0} & M_a \end{pmatrix}$$

And, $\eta'_F = \begin{pmatrix} 0 \\ \eta_F \end{pmatrix}$. Now, it is obvious that, for any $w = a_1 a_2 \cdots a_n \in \Sigma^*$,

$$\begin{aligned} \mathcal{P}_R(w) &= (1, \mathbf{0}) M'_\epsilon M'_{a_1} M'_{a_2} \cdots M'_{a_n} \eta'_F \\ &= (1, \mathbf{0}) \begin{pmatrix} 0 & \pi M_{a_1} M_{a_2} \cdots M_{a_n} \\ 0 & M_{a_1} M_{a_2} \cdots M_{a_n} \end{pmatrix} \begin{pmatrix} 0 \\ \eta_F \end{pmatrix} \\ &= \pi M_{a_1} M_{a_2} \cdots M_{a_n} \eta_F \\ &= \mathcal{P}_{\mathcal{A}}(w), \end{aligned}$$

as required. □

The assignment ν described in Definition 3.2 should be modified slightly, i.e., for any configuration (q, n), whenever $q = q'$, $\nu(q, n) = \nu(q', n')$. Then the proof of Theorem 1.2, which will be omitted here, follows from the above Lemma 4.1 and Theorem 1.1.

Remark 4. *The definition of probabilistic one-counter automata here is different from the one in [12]. Now let $|\Sigma| = 1$, we get a special case of probabilistic one-counter automata, which we name it the 'probabilistic one-counter process'. It is clear that given a probabilistic one-counter process, it induces an infinite Markov chain. Hence, the proof method to Theorem 1.2 may not be applicable for this case. We conjecture the corresponding model-checking problem is decidable, based on that it is difficult to encode "Post Corresponding Problem" to its transition rules. Furthermore, the method to its decidability is still lack, which means this problem may be more difficult.*

5 Conclusions

In this paper, we studied the problems of model-checking probabilistic branching-time properties of probabilistic automata and probabilistic one-counter automata. The outcome shows that these problems are undecidable. Meanwhile, the model-checking problem of probabilistic one-counter process remains open, which is the future work.

References

[1] A.N. Shiryaev, Probability, (Second Edition), Springer-Verlag, New York, 1995.

[2] A. Paz, Introduction to the probabilistic automata, Academic Press, New York, 1971.

[3] C. Baier, and J.P. Katoen, Principles of Model Checking, MIT Press, 2008.

[4] C. Baier, N. Bertrand, and P. Schnoebelen, A note on the attractor-property of infinite-state Markov chains, Information Processing Letters 97 (2006) 58-63.

[5] D.J. White, Markov Decision Processes, John Wiley, 1993.

[6] E.L. Post, A variant of a recursively unsolvable problem, Bulletin of the American Mathematical Society 52, 1946, pp. 264-268.

[7] E.M. Clarke, O. Grumberg, and D.A. Peled, Model Checking, MIT Press, 1999.

[8] H. Hansson, and B. Jonsson, A logic for reasoning about time and reliability, Formal Aspects of Computing 6 (1994) 512-535.

[9] J.W. Carlyle, Reduced forms for stochastic sequential machines, Journal Mathematical Analysis Application 7 (1963) 167-175.

[10] J.E. Hopcroft, and J.D. Ullman, Formal languages and their relation to automata, Addison-Wesley, 1969.

[11] J. Esparza, A. Kučera, and R. Mayr, Model checking probabilistic pushdown automata, Logical Methods in Computer Science, Vol. 2 (1:2) 2006, pp. 1-31.

[12] K. Etessami, D. Wojtczak, M. Yannakakis, Quasi-Birth-Death Processes, Tree-Like QBDs, Probabilistic 1-Counter Automata, and Pushdown Systems, in: $5th$ Int. Conf. on Quantitative Evaluation of Systems (QEST'08), 2008

[13] L.G. Valiant, and M.S. Paterson, Deterministic One-Counter Automata, Journal of Computer and System Sciences, 10 (1975) 340-350.

[14] M. Loève, Probability Theory I (4th edtion), Springer-Verlag, New York, 1978.

[15] M. Loève, Probability Theory II (4th edtion), Springer-Verlag, New York, 1978.

[16] M. Kwiatkowska, G. Norman, J. Sproston, and F. Wang, Symbolic model checking for probabilistic timed automata, Information and Computation 25 (2007) 1027-1077.

[17] M.O. Rabin, Probabilistic Automata, Information and Control 6, 1963, pp. 230-245.

[18] T. Brázdil, V. Brožek, V. Forejt, and A. Kučera, Branching-time model-checking of probabilistic pushdown automata, Journal of Computer and System Sciences 80 (2014) 139-156.